Application Note
Using Serial I/O with Shared Interrupts under Linux

Date: 04/30/2004

Applies to WinSystems Products:
PCM-COM4A and PCM-COM8

Issue Addressed:
How to configure Linux to work with the shared interrupt mode on the PCM-COM4A and PCM-COM8

Solution:
Starting with kernel version 2.2, Linux offers built-in support for at least some standard serial boards with shared interrupt capability. The kernel must have been built with the "Extended dumb serial driver options" enabled. (See the Serial-HOWTO for additional information) Make sure that your kernel build includes these features before proceeding with the following configuration steps.

First, verify that the required device files (`ttyS#`) exist in your `/dev` directory. You will probably have `ttyS0` (a.k.a. COM1) through `ttyS3` (a.k.a. COM4) but no others. If this is so, use `MAKEDEV` to create additional device files, one for each serial port (beyond COM1-COM4) that you will have in your system. If you have one PCM-COM8 installed, you will want to add devices `ttyS4` through `ttyS11`, at a minimum. An example follows:

```
linux# cd /dev
linux# ./MAKEDEV ttyS4
linux# ./MAKEDEV ttyS5
.
.
.
linux# ./MAKEDEV ttyS11
```

Next, create or modify the `rc.serial` file to include the commands from whichever of the following examples is appropriate:

PCM-COM8 Example

```
#!/bin/sh
setserial /dev/ttyS4  port 0x138 irq 7 uart 16550A ^fourport
setserial /dev/ttyS5  port 0x140 irq 7 uart 16550A ^fourport
setserial /dev/ttyS6  port 0x148 irq 7 uart 16550A ^fourport
setserial /dev/ttyS7  port 0x150 irq 7 uart 16550A ^fourport
setserial /dev/ttyS8  port 0x158 irq 7 uart 16550A ^fourport
setserial /dev/ttyS9  port 0x160 irq 7 uart 16550A ^fourport
setserial /dev/ttyS10 port 0x168 irq 7 uart 16550A ^fourport
setserial /dev/ttyS11 port 0x170 irq 7 uart 16550A ^fourport
setserial /dev/ttyS4 set_multiport port1 0x133 mask1 0xff match1 0
```

Notes
This example is for a PCM-COM8 with a base address of 0x130 and set to use interrupt 7. The use of the `^fourport` *("not fourport") switch option may seem counterintuitive but the* `fourport` *option only applies to AST Fourport boards which function differently than WinSystems multiport boards. You may notice that the uart specified in this example does not take advantage of the deep FIFOs that are available on the PCM-COM8. This is to maintain consistency between the provided examples and because the setserial version we tested with did not support 128 Byte FIFOs.*

PCM-COM4A Example

```
#!/bin/sh
setserial /dev/ttyS4 port 0x100 irq 7 uart 16550A ^fourport
setserial /dev/ttyS5 port 0x108 irq 7 uart 16550A ^fourport
setserial /dev/ttyS6 port 0x110 irq 7 uart 16550A ^fourport
setserial /dev/ttyS7 port 0x118 irq 7 uart 16550A ^fourport
setserial /dev/ttyS4 set_multiport port1 0x240 mask1 0xf match1 0
```

Notes
This example is for a PCM-COM4A configured for Map #4 and set to use interrupt 7. The `mask1` value on the last line differs from the PCM-COM8 board because there are only 4 status bits to check. The use of the `^fourport` (not fourport) switch option may seem counterintuitive but the `fourport` option only applies to AST Fourport boards which function differently than WinSystems multiport boards.

Finally, if you had to create rc.serial, make sure it is executable and in the proper directory so that it executes whenever you boot your system. To make the file executable, run:

```
linux# chmod +x rc.serial
```

Place `rc.serial` in the same directory as `rc.local` which should be resident on most systems.

Reboot so that Linux initializes the ports and you're done. If you want to check to be sure that the ports were initialized properly, run `setserial` with the `-a` option to report the status of each port. For example:

```
setserial -a /dev/ttyS4
```

Also check to make sure that the multiport settings are correct:

```
setserial /dev/ttyS4 get_multiport
```

Setserial's output should match your hardware configuration and the settings you placed in `rc.serial.` To verify that the ports are functional, you may try something simple like using `echo` to send a text string to another machine running a terminal program. Another simple test is to use `minicom` to send/receive files sent from another machine.

For additional information about multiport serial I/O, see the Serial-HOWTO available at http://www.tldp.org. For information on `minicom`, run `man minicom`..

Tested Version(s):
Red Hat Linux 6.2, Setserial 2.15, Minicom 1.83.0

If you have questions concerning this Application Note, please contact your Applications Engineer at http://www.winsystems.com/sales/contacts1.html.